Functor (1A)

Young Won Lim
7/4/17

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice.

Based on

http://learnyouahaskell.com/making-our-own-types-and-typeclasses#the-functor-typeclass

Haskell in 5 steps
https://wiki.haskell.org/Haskell_in_5_steps

Maybe as a functor

Functor type class:

- transforming one type to another
- transforming operations of one type to those of another

Maybe a has a useful instance of a **functor** type class

Functor provides **fmap** method

maps functions of the base type (such as Integer)
to *functions* of the lifted type (such as Maybe Integer).

https://stackoverflow.com/questions/18808258/what-does-the-just-syntax-mean-in-haskell

Maybe as a functor

A *function* **f** transformed with **fmap**
cab work on a Maybe value

```
case maybeVal of
  Nothing  -> Nothing       -- there is nothing, so just return Nothing
  Just val -> Just (f val)  -- there is a value, so apply the function to it
```

```
   father :: Person -> Maybe Person
   mother :: Person -> Maybe Person
```

```
       f :: Int                -> Int
fmap   f :: Maybe Integer -> Maybe Integer
```

a **Maybe Integer** value: **m_x**

```
fmap  f   m_x
```

https://stackoverflow.com/questions/18808258/what-does-the-just-syntax-mean-in-haskell

Maybe as a functor

In fact, you could apply a whole chain of
lifted Integer -> **Integer** functions to **Maybe Integer** values
and only have to worry about explicitly checking for **Nothing** once when you're finished.

https://stackoverflow.com/questions/18808258/what-does-the-just-syntax-mean-in-haskell

Typeclasses

Typeclasses are like interfaces

defines some behavior
- comparing for equality
- comparing for ordering
- enumeration

Instances of that typeclass
- types possessing such behavior

Such behavior is defined by
- **function definition**
- **type declaration** to be implemented

a type is an instance of a typeclass implies
the functions defined by the typeclass with that type can be used

No relation with classes in Java or Python

http://learnyouahaskell.com/making-our-own-types-and-typeclasses#the-functor-typeclass

A Typeclass Example

the Eq typeclass

defines the functions **==** and ***/=***

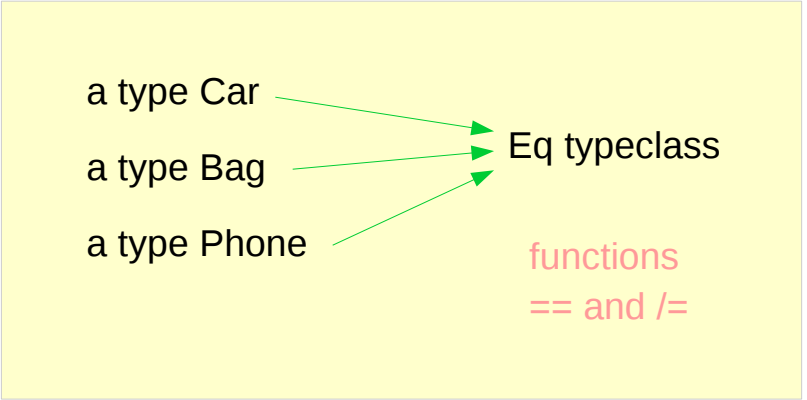
a type Car

comparing two cars c1 and c2 with the equality function ==

The Car type is an instance of Eq typeclass

Instances : various types

Typeclass : a group or a class of these similar types



http://learnyouahaskell.com/making-our-own-types-and-typeclasses#the-functor-typeclass

Eq Typeclass Example

```
class Eq a where
   (==) :: a -> a -> Bool          - a type declaration
   (/=) :: a -> a -> Bool          - a type declaration
   x == y = not (x /= y)           - a function definition
   x /= y = not (x == y)           - a function definition
```

```
data TrafficLight = Red | Yellow | Green
```

```
instance Eq TrafficLight where
   Red    == Red    = True
   Green  == Green  = True
   Yellow == Yellow = True
   _      == _      = False
```

```
ghci> Red == Red
True
ghci> Red == Yellow
False
ghci> Red `elem` [Red, Yellow, Green]
True
```

http://learnyouahaskell.com/making-our-own-types-and-typeclasses#the-functor-typeclass

Show Typeclass Example

```
class Show  a where
   show :: a  -> String              - a type declaration
   * * *
```

```
data TrafficLight = Red | Yellow | Green

instance Show TrafficLight where
    show Red = "Red light"
    show Yellow = "Yellow light"
    show Green = "Green light"
```

```
ghci> [Red, Yellow, Green]
[Red light,Yellow light,Green light]
```

http://learnyouahaskell.com/making-our-own-types-and-typeclasses#the-functor-typeclass

Show Typeclass Example

```
class (Eq a) => Num a where
  ...

class Num a where
  ...
```

class constraint on a class declaration
only we state that our type a must be an instance of **Eq**

we have to make a type an instance of **Eq**
before we can make it an instance of **Num**

When defining function bodies
in the class declaration or
in instance declarations,

we can safely use == because a is a part of **Eq**

http://learnyouahaskell.com/making-our-own-types-and-typeclasses#the-functor-typeclass

References

[1] ftp://ftp.geoinfo.tuwien.ac.at/navratil/HaskellTutorial.pdf
[2] https://www.umiacs.umd.edu/~hal/docs/daume02yaht.pdf